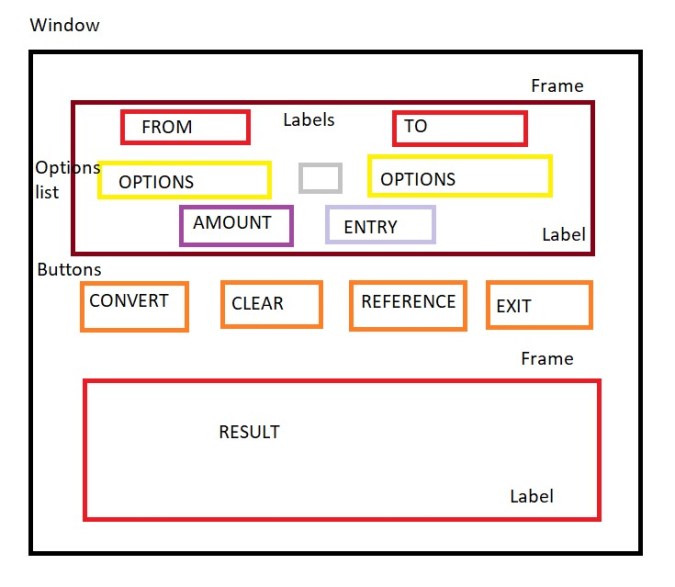
**[Currency Converter Application in Python](https://copyassignment.com/currency-converter-desktop-application-in-python/)**



## What we’ll do?

We are coding for a program that can convert currencies at the current time, as currency between different countries trades over a second, hence this will be a ‘REAL-TIME CONVERTER’.  
We will design this with GUI for a better view of our program and then convert it into an application! Yes, a desktop-based application!

## Modules we need:

For designinng : tkinter  
For Currency Conversion: forex-python  
For converting to exe: Pyinstaller (auto-py-to-exe)

## **Introduction to forex-python:**

forex-python is a python module that provides functionality for foreign exchange rates and currency conversion. It can give historical rates for any day since the year 1999. It also decodes currency names and symbols.  
  
This module comes with PyPI, so it can be installed through pip in cmd:

pip install forex-python

It not only gives Currency rates but Bitcoins as well!  
For currency, we need to first import the rates and create the instance of the same class.

**from forex\_python.converter import CurrencyRates**

**c = CurrencyRates() #creating instance**

**r = c.convert("USD", "INR", 1)**

**print(r)**

The above function takes ‘from’, ‘to’ and ‘amount’ as parameters and return real-time conversion for the specified amount. It takes ‘from’ and ‘to’ as the abbreviated code of currency names of different regions.

**Output:**

72.867501648

Also, in our project, we will provide an option as bitcoin conversion.  
For the sake of it, we need to import bitcoin rates and creating its instance as above separately.

**from forex\_python.bitcoin import BtcConverter**

**b = BtcConverter()**

**r1 = b.convert\_to\_btc(100, "USD") #converting 100 USD into bitcoins**

**r2 = b.convert\_btc\_to\_cur(1, "USD") #converting 1 bitcoins into USD**

**print(r1, r2, end="\n")**r1 will give us the amount in specified currency into bitcoins whereas r2 will set the value of the specified amount of bitcoin into the specified currency.  
Note: Don’t get confused, the name of functions itself suggest their works.

**Output:**

0.0029284254773098524 #amount in BTC = 100 USD

34161.4683 #amount in USD = 1 BTC

These are the functions that we’ll use in our code. forex\_python also provides standard rates in all currencies provided one currency, symbols, abbreviations, etc.  
For more information about it, read the documentation:<https://forex-python.readthedocs.io/en/latest/usage.html>

Now,  
With the hope, that you are cleared over the central logic of our code, let’s get started with designing:  
We will make this Currency Converter Desktop Application in Python a GUI, so obviously we’ll need GUI tools and for it here we are using tkinter. You can also use wxPython or PyQt, etc.  
I am making this application a little bit eye-catchy, by adding background images, and a logo image as a conversion sign.

We will create two different frames, one for our main widgets, labels, and options menu providing options for selection of currency (including bitcoins), a label and entry for amount, a label for logo image to be placed between ‘FROM’ and ‘TO’ for a catchy look.

The other frame will contain a label displaying result.

Between these two frames, there are four buttons:  
**CONVERT:** For displaying the result.  
**CLEAR:**To clear the AMOUNT and RESULT entries.  
**REFERENCE:** This is a little different, as this button will show the full form of abbreviations used in the options menu for help/reference purposes. The most interesting thing about this, that we will redirect the user to another window that will display the full forms.  
**EXIT:**To quit the application.

This is it!  
Now less discussing, more coding:  
Let’s start by creating the design:  
Importing modules for designing:

**from tkinter import \***

**from PIL import ImageTk, Image**

**import tkinter.font as font**

As we’ve discussed we’ll add a background image. Here, I m using the ‘jpg’ image, and tkinter’s label widget doesn’t support ‘jpg’s hence, here we’re using PIL. tkinters’s font is also imported to change the font style.

**root = Tk()**

**root.title("REAL TIME CURRENCY CONVERTOR")**

**#FIXED SIZE OF WINDOW**

**root.minsize(600,500)**

**root.maxsize(600,500)**

**HEIGHT = 500**

**WIDTH = 500**

**FONT = font.Font(family ="Comic Sans MS", size ="9", weight ="bold")**

**canvas = Canvas(root, height = HEIGHT, width = WIDTH)**

**canvas.pack()**

**background\_image = ImageTk.PhotoImage(Image.open(r"FULL PATH.ext"))**

**background\_label = Label(root, image = background\_image)**

**background\_label.place(relwidth = 1, relheight =1)**

**frame = Frame(root, bg ="yellow", bd =5)**

**frame.place(relx = 0.5, rely = 0.1, relwidth = 0.80, relheight = 0.25, anchor = "n")**

**label\_up = Label(frame)**

**label\_up.place( relwidth= 1 , relheight = 1)**

**lower\_frame = Frame(root, bg ="yellow", bd =10)**

**lower\_frame.place(relx = 0.5, rely = 0.53, relwidth = 0.8, relheight = 0.25, anchor = "n")**

**label\_down = Label(lower\_frame, font = FONT, fg = "#001a4d", anchor = "nw", justify = "left", bd =4)**

**label\_down.place( relwidth=1, relheight = 1)**

**root.mainloop()**

Because of minsize and maxsize, the size of the window will be fixed which will avoid stretching of widgets while minimizing and maximizing.  
We’ve used the ‘place’ method for placing. You can use a pack or grid.

Now, let’s add labels and buttons, also that tiny conversion image. For a list of options, we’ll use the OptionMenu widget.  
Also, we’ll give commands to the buttons and write the functions next.

**label1 = Label(frame, text = "FROM", font =FONT, bd =5, bg ="#d9138a", highlightbackground = "#d9138a", fg = "white")**

**label1.place(relx = 0.15, rely = 0.02,relwidth = 0.15, relheight =0.25)**

**label2 = Label(frame, text = "TO", font =FONT, bd =5, bg ="#d9138a", highlightbackground = "#d9138a", fg = "white")**

**label2.place(relx = 0.64,rely = 0.03,relwidth = 0.15, relheight =0.25)**

**#for OptionMenu**

**options = [**

**"BTC",**

**"USD",**

**"EUR",**

**"JPY",**

**"GBP",**

**"AUD",**

**"CAD",**

**"CHF",**

**"INR",**

**"RUB",**

**"CNY"**

**]**

**clicked1 = StringVar()**

**clicked1.set("Select")**

**listbox1 = OptionMenu(frame,clicked1, \*options)**

**listbox1.config(bg = "#fc034e", fg = "black", activeforeground = "#fc034e", activebackground = "black", font=FONT)**

**listbox1.place(relx = 0.07,rely = 0.3, relheight = 0.28, relwidth = 0.38)**

**clicked2 = StringVar()**

**clicked2.set("Select")**

**listbox2 = OptionMenu(frame,clicked2, \*options)**

**listbox2.config(bg = "#fc034e", fg = "black", activeforeground = "#fc034e", activebackground = "black", font=FONT)**

**listbox2.place(relx = 0.56,rely = 0.3, relheight = 0.28, relwidth = 0.38)**

**#for logo image between two options list**

**image = PhotoImage(file = r"FULL PATH.ext")**

**img\_label = Label(frame, image = image)**

**img\_label.place(relx = 0.445, rely = 0.22)**

**#amount**

**label3 = Label(frame, text = "AMOUNT", font = FONT, bg = "#12a4d9", highlightbackground = "#12a4d9", fg = "white")**

**label3.place(relx = 0.26,rely = 0.7,relwidth = 0.2, relheight = 0.25)**

**entry = Entry(frame, font = FONT, fg = "#001a4d")**

**entry.place(relx = 0.54, rely = 0.7, relwidth=0.26, relheight = 0.25)**

**#buttons**

**button1 = Button(root, text = "CONVERT", font = FONT, bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black")**

**button1.place(relx = 0.16,rely = 0.4,relwidth = 0.15, relheight = 0.07)**

**button2 = Button(root, text = "CLEAR", font = FONT, bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black")**

**button2.place(relx = 0.35,rely = 0.4,relwidth = 0.13, relheight = 0.07)**

**button3 = Button(root, text = "REFERENCE", font = FONT, bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black")**

**button3.place(relx = 0.52, rely = 0.4, relwidth = 0.15, relheight = 0.07)**

**button4= Button(root, text = "EXIT", font = FONT, bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black")**

**button4.place(relx = 0.7, rely = 0.4, relwidth = 0.12, relheight = 0.07)**

There’s not any rocket science we’ve used here.  
Yes, clicked1 and clicked2 are the two variables we’ve created to get the value of currency the user will select. You can increase the number of options in the list. And used the get() function to get the value, the user is selecting.  
Note:  
Here, that image symbolizing conversion, in my case, was of ‘.png’ type and hence I’ve used PhtoImage() of tkinter and not as we’ve done earlier for ‘.jpg’ or ‘.jpeg’ extension. You can also use the same for ‘.png’ kind of.

Now, let’s add the commands to buttons:  
**1. CONVERT:**  
In this function, first, we’ll check if all entries (both option menus, amount) are filled or not, whether the amount entered by the user is valid or not. Any case of invalidity will raise an error shown by the MessageBox widget.  
And as we’ve discussed earlier if the user selects, BTC (bitcoin), we’ll require another function for it.  
CONVERT will take three arguments, two currency args selected and amount. As this is a clickable event, we will command it using the lambda function.

**from tkinter import messagebox**

**from forex\_python.converter import CurrencyRates**

**from forex\_python.bitcoin import BtcConverter**

**def convert(c1,c2,amount):**

**try:**

**if amount == "":**

**messagebox.showerror("Error", "Amount not specified")**

**elif c1 == "Select" or c2 == "Select":**

**messagebox.showinfo("Error", "Currency not selected")**

**else:**

**try:**

**amount = float(amount)**

**b = BtcConverter()**

**c = CurrencyRates()**

**if c1 == c2:**

**result = amount**

**elif c1 == "BTC":**

**result = b.convert\_btc\_to\_cur(amount, c2)**

**elif c2 == "BTC":**

**result = b.convert\_to\_btc(amount, c1)**

**else:**

**result = c.convert(c1, c2, int(amount))**

**print(result)**

**label\_down["text"] = f"Conversion Result: {amount} {c2}\n{amount} {c1} = {result} {c2}"**

**except ValueError:**

**messagebox.showerror("Error", "Invalid amount")**

**clear()**

**except Exception:**

**messagebox.showerror("Error", "Something went wrong. Please try again")**

**button1["command"] =lambda:convert(clicked1.get(), clicked2.get(), entry.get())**

You can also specify ‘command’ by the time declaring button, but, here, we’re seeing this in chunks of code, we’ve specified it separately.

NOTE:  
If the user enters the amount as ‘6.78’ while converting to ‘int’ in conversion calculations, the python interpreter will treat ‘.’ as a special character and not a floating decimal. Hence, instead of converting it to ‘int’, we’ll choose ‘float’.

**2. CLEAR:**  
The CLEAR function will be clear all the entries i.e. amount entry. Label result also needs to be clear if the user converts any currency and if he again wants to convert, the label must be clear.

**def clear():**

**entry.delete(0,END)**

**label\_down["text"] = ""**

**button2["command"] = clear**

**3. REFERENCE:**  
As we discussed earlier, this function is for giving info about currency abbreviations i.e. its full form.  
We will show this info in the new window and also add the ‘BACK’ button to go back to the main window.  
You can also show this to the result label.

**def help():**

**newwin = Tk()**

**newwin.title("Reference")**

**newwin.maxsize(400,300)**

**newwin.minsize(400,300)**

**newcanvas = Canvas(newwin, height = 400, width = 300)**

**newcanvas.pack()**

**newframe = Frame(newwin, bg ="yellow")**

**newframe.place(relwidth = 1, relheight = 1)**

**newlabel = Label(newframe, font = ("Comic Sans MS", 11, "bold"), fg ="#001a4d", anchor = "nw", justify = "left", bd =4)**

**newlabel.place(relx = 0.05, rely = 0.05,relwidth = 0.90, relheight = 0.90)**

**newlabel["text"] = "Abbrevations:\nBTC - Bitcoin\nUSD - USD Dollar\nEUR - Euro\nJPY - Japnese Yen\nGBP - Pound Sterling\nAUD - Australian Dollar\nCAD - Canadian Dollar\nCHF - Swiss Frank\nINR - Indian Rupees\nRUB - Russian Rubble\nCNY - Chinese Yuan"**

**newbutton = Button(newframe, text = "Back",font = ("Comic Sans MS", 11, "bold"), bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black", command = lambda:newwin.destroy())**

**newbutton.place(relx = 0.76, rely = 0.82, relwidth = 0.14, relheight = 0.11)**

**newwin.mainloop()**

**button3["command"] = help**

**4. EXIT:**  
The EXIT button will simply destroy the root window to end the program.

**def exit():**

**root.destroy()**

**button["command"] = exit**

here’s the whole code

**from tkinter import \***

**from tkinter import messagebox**

**from PIL import ImageTk, Image**

**import tkinter.font as font**

**from forex\_python.converter import CurrencyRates**

**from forex\_python.bitcoin import BtcConverter**

**root = Tk()**

**root.title("REAL TIME CURRENCY CONVERTOR")**

**root.minsize(600,500)**

**root.maxsize(600,500)**

**HEIGHT = 500**

**WIDTH = 500**

**FONT = font.Font(family ="Comic Sans MS", size ="9", weight ="bold")**

**#functions**

**def clear():**

**entry.delete(0,END)**

**label\_down["text"] = ""**

**def convert(c1,c2,amount):**

**try:**

**if amount == "":**

**messagebox.showerror("Error", "Amount not specified")**

**elif c1 == "Select" or c2 == "Select":**

**messagebox.showinfo("Error", "Currency not selected")**

**else:**

**try:**

**amount = float(amount)**

**b = BtcConverter()**

**c = CurrencyRates()**

**if c1 == c2:**

**result = amount**

**elif c1 == "BTC":**

**result = b.convert\_btc\_to\_cur(amount, c2)**

**elif c2 == "BTC":**

**result = b.convert\_to\_btc(amount, c1)**

**else:**

**result = c.convert(c1, c2, int(amount))**

**print(result)**

**label\_down["text"] = f"Conversion Result: {amount} {c2}\n{amount} {c1} = {result} {c2}"**

**except ValueError:**

**messagebox.showerror("Error", "Invalid amount")**

**clear()**

**except Exception:**

**messagebox.showerror("Error", "Something went wrong. Please try again")**

**def help():**

**newwin = Tk()**

**newwin.title("Reference")**

**newwin.maxsize(400,300)**

**newwin.minsize(400,300)**

**newcanvas = Canvas(newwin, height = 400, width = 300)**

**newcanvas.pack()**

**newframe = Frame(newwin, bg ="yellow")**

**newframe.place(relwidth = 1, relheight = 1)**

**newlabel = Label(newframe, font = ("Comic Sans MS", 11, "bold"), fg ="#001a4d", anchor = "nw", justify = "left", bd =4)**

**newlabel.place(relx = 0.05, rely = 0.05,relwidth = 0.90, relheight = 0.90)**

**newlabel["text"] = "Abbrevations:\nBTC - Bitcoin\nUSD - USD Dollar\nEUR - Euro\nJPY - Japnese Yen\nGBP - Pound Sterling\nAUD - Australian Dollar\nCAD - Canadian Dollar\nCHF - Swiss Frank\nINR - Indian Rupees\nRUB - Russian Rubble\nCNY - Chinese Yuan"**

**newbutton = Button(newframe, text = "Back",font = ("Comic Sans MS", 11, "bold"), bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black", command = lambda:newwin.destroy())**

**newbutton.place(relx = 0.76, rely = 0.82, relwidth = 0.14, relheight = 0.11)**

**newwin.mainloop()**

**def exit():**

**root.destroy()**

**canvas = Canvas(root, height = HEIGHT, width = WIDTH)**

**canvas.pack()**

**background\_image = ImageTk.PhotoImage(Image.open(r"FULL PATH.ext"))**

**background\_label = Label(root, image = background\_image)**

**background\_label.place(relwidth = 1, relheight =1)**

**frame = Frame(root, bg ="yellow", bd =5)**

**frame.place(relx = 0.5, rely = 0.1, relwidth = 0.80, relheight = 0.25, anchor = "n")**

**label\_up = Label(frame)**

**label\_up.place( relwidth= 1 , relheight = 1)**

**label1 = Label(frame, text = "FROM", font =FONT, bd =5, bg ="#d9138a", highlightbackground = "#d9138a", fg = "white")**

**label1.place(relx = 0.15, rely = 0.02,relwidth = 0.15, relheight =0.25)**

**options = [**

**"BTC",**

**"USD",**

**"EUR",**

**"JPY",**

**"GBP",**

**"AUD",**

**"CAD",**

**"CHF",**

**"INR",**

**"RUB",**

**"CNY"**

**]**

**clicked1 = StringVar()**

**clicked1.set("Select")**

**listbox1 = OptionMenu(frame,clicked1, \*options)**

**listbox1.config(bg = "#fc034e", fg = "black", activeforeground = "#fc034e", activebackground = "black", font=FONT)**

**listbox1.place(relx = 0.07,rely = 0.3, relheight = 0.28, relwidth = 0.38)**

**label2 = Label(frame, text = "TO", font =FONT, bd =5, bg ="#d9138a", highlightbackground = "#d9138a", fg = "white")**

**label2.place(relx = 0.64,rely = 0.03,relwidth = 0.15, relheight =0.25)**

**image = PhotoImage(file = r"FULL PATH.ext")**

**img\_label = Label(frame, image = image)**

**img\_label.place(relx = 0.445, rely = 0.22)**

**clicked2 = StringVar()**

**clicked2.set("Select")**

**listbox2 = OptionMenu(frame,clicked2, \*options)**

**listbox2.config(bg = "#fc034e", fg = "black", activeforeground = "#fc034e", activebackground = "black", font=FONT)**

**listbox2.place(relx = 0.56,rely = 0.3, relheight = 0.28, relwidth = 0.38)**

**label3 = Label(frame, text = "AMOUNT", font = FONT, bg = "#12a4d9", highlightbackground = "#12a4d9", fg = "white")**

**label3.place(relx = 0.26,rely = 0.7,relwidth = 0.2, relheight = 0.25)**

**entry = Entry(frame, font = FONT, fg = "#001a4d")**

**entry.place(relx = 0.54, rely = 0.7, relwidth=0.26, relheight = 0.25)**

**button1 = Button(root, text = "CONVERT", font = FONT, bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black", command = lambda:convert(clicked1.get(), clicked2.get(), entry.get()))**

**button1.place(relx = 0.16,rely = 0.4,relwidth = 0.15, relheight = 0.07)**

**button2 = Button(root, text = "CLEAR", font = FONT, bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black", command = clear)**

**button2.place(relx = 0.35,rely = 0.4,relwidth = 0.13, relheight = 0.07)**

**button3 = Button(root, text = "REFERENCE", font = FONT, bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black", command = help)**

**button3.place(relx = 0.52, rely = 0.4, relwidth = 0.15, relheight = 0.07)**

**button4= Button(root, text = "EXIT", font = FONT, bg = "pink", fg = "black", activeforeground = "pink", activebackground = "black", command = exit)**

**button4.place(relx = 0.7, rely = 0.4, relwidth = 0.12, relheight = 0.07)**

**lower\_frame = Frame(root, bg ="yellow", bd =10)**

**lower\_frame.place(relx = 0.5, rely = 0.53, relwidth = 0.8, relheight = 0.25, anchor = "n")**

**FONT = font.Font(family ="Comic Sans MS", size ="12", weight ="bold")**

**label\_down = Label(lower\_frame, font = FONT, fg = "#001a4d", anchor = "nw", justify = "left", bd =4)**

**label\_down.place( relwidth=1, relheight = 1)**

**root.mainloop()**